METHOD AND SYSTEM FOR SAFELY ARBITRATING DISK DRIVE OWNERSHIP

FIELD OF THE INVENTION

The present invention is related to a data storage system. More specifically, the present invention pertains to a data storage system with D disks, where each disk has a reserved disk block for each of N servers that uses a timestamp-based voting algorithm to arbitrate access of the servers to a set of disks, or each server reads all the other servers' disk blocks in each disk in order to determine which server has access to, and use and control of the disks at a given time.

BACKGROUND OF THE INVENTION

In many environments requiring a highly available system, disk drives need to be shared among two or more machines, such that if one machine fails, one of the others can provide access to the data stored on the shared disk drives. In such environments, it is crucial that the data on the disk drives not be accessed by multiple machines simultaneously, to prevent serious data corruption. The present invention provides for an arbitration that has fewer failure modes than previous systems for shared disk drives.

There are two areas in which the present invention is superior to the prior art in the area of disk ownership arbitration.

Typical prior art for arbitration among N servers for access to a set of disks involves a heartbeat being exchanged between two servers, and when the primary server crashes, the secondary server detects a loss of heartbeat and takes over for the

failed primary. There are three weaknesses with this art that are solved by this invention.

First, the prior art typically works only with 2 servers, rather than an arbitrary number of servers.

Second, by using the disk as the communications mechanism, rather than a separate communication path, the present invention reduces the chances that the communication path used for determining which server should access the disk will fail independently from the communication path used for accessing the disk itself.

Third, even with a network partition, it is guaranteed that at most one server is granted access to the arbitrated set of disks, while most prior art has the possibility of assigning two servers access to the set of disks in the case where there is a network partition in the communications network, and both servers are actually still connected to the disks.

Another prior art device (Ubik, by Transarc Corporation [1989]), used a similar voting mechanism, run over an IP network, to elect a synchronization server for a distributed database. However, the Ubik system did not use an array of disk blocks for a communication medium, but instead used network packets exchanged over an IP network.

SUMMARY OF THE INVENTION

The present invention pertains to a data storage system. The system comprises N servers, where $N \geq 2$ and is an integer. The system comprises D disks, where $D \geq 2$ and is an integer. Each server is in communication with each disk. Each disk has a

reserved disk block for each of the N servers. The system comprises a disk arbitration mechanism that uses a timestamp-based voting algorithm over the disk blocks associated with the servers to exchange votes for a primary server to arbitrate access of the servers to a set of disks.

The present invention pertains to a data storage system. The system comprises N servers, where $N \geq 2$ and is an integer. The system comprises D disks, where $D \geq 2$ and is an integer. Each server is in communication with each disk. Each disk has a reserved disk block for each of the N servers. The system comprises a disk arbitration mechanism where each of the N servers writes its state in its own associated disk block in each disk, and reads all the other servers' disk blocks in each disk in order to determine which server has access to, and use and control of the disks at a given time.

The present invention pertains to a method for storing data. The method comprises the steps of writing by N servers into each servers' own associated disk block in each disk of D disks its state, where $N \geq 2$ and $D \geq 2$ and are integers. There is the step of reading by each server all the other servers' disk blocks in each disk in order to determine which server has access to, and use and control of the disks at a given time.

BRIEF DESCRIPTION OF THE DRAWINGS

In the accompanying drawings, the preferred embodiment of the invention and preferred methods of practicing the invention are illustrated in which:

Figure 1 is a schematic representation of a system of the present invention.

Figure 2 is a schematic representation of the system topology.

Figure 3 is a schematic representation of the system which shows which disk blocks are written and read by which servers in a three server configuration.

Figure 4 shows the states that a server participating in the election protocol goes through.

DETAILED DESCRIPTION

Referring now to the drawings wherein like reference numerals refer to similar or identical parts throughout the several views, and more specifically to figure 1 thereof, there is shown a data storage system 10. The system 10 comprises N servers 12, where $N \geq 2$ and is an integer. The system 10 comprises D disks 14, where $D \geq 2$ and is an integer. Each server 12 is in communication with each disk. Each disk has a reserved disk block 18 for each of the N servers 12. The system 10 comprises a disk arbitration mechanism 16 that uses a timestamp-based voting algorithm over the disk blocks 18 associated with the servers 12 to exchange votes for a primary server to arbitrate access of the servers 12 to a set of disks 14.

The present invention pertains to a data storage system 10, as shown in figure 1. The system 10 comprises N servers 12, where $N \geq 2$ and is an integer. The system 10 comprises D disks 14, where $D \geq 2$ and is an integer. Each server 12 is in communication with each disk. Each disk has a reserved disk block 18 for each of the N servers 12. The system 10 comprises a disk arbitration mechanism 16 where each of the N servers 12 writes its state in its own associated disk block 18 in each disk, and reads all the other

servers' 12 disk blocks 18 in each disk in order to determine which server 12 has access to, and use and control of the disks 14 at a given time.

Preferably, each server 12 has an index. The disk arbitration mechanism 16 preferably causes each server 12 at first predetermined times to read all of the disk blocks 18, and write its own disk block 18 to determine which server 12 has access to, and use and control of the disks 14 at a given time.

Preferably, each server 12 includes a state machine 20 and a local RAM 22, and maintains in local RAM 22 a last time at which each servers' 12 state's heartbeat counter changed and a value associated with the state when it last changed. Each server 12 preferably determines which of the other servers 12 are operating by identifying which of the other servers 12 had their state's heartbeat counter change during second predetermined times.

The present invention pertains to a method for storing data. The method comprises the steps of writing by N servers 12 into each servers' 12 own associated disk block 18 in each disk of D disks 14 its state, where $N \geq 2$ and $D \geq 2$ and are integers. There is the step of reading by each server 12 all the other servers' 12 disk blocks 18 in each disk in order to determine which server 12 has access to, and use and control of the disks 14 at a given time.

Preferably, the reading step includes the step of performing a voting protocol to determine which server 12 has access to, and use and control of the disks 14 at a given time. After the reading step, there are preferably the steps of determining which server 12 becomes a winning server 12 and has access to, and use and control of the disk at a given time; and accessing the disk exclusively by the winning server 12.

Preferably, the accessing step includes the step of transmitting by the winning server 12 its state from not winning to winning and invalidating by the winning server 12 all caches of the winning server 12.

The writing step preferably includes the step of assigning an index to each server 12 only at initialization of the servers 12 and the disks. Preferably, the reading step includes the step of reading at predetermined times by each server 12 all disk blocks 18. The writing step preferably includes the step of writing its own respective disk block 18. The writing step preferably includes the step of maintaining by each server 12 in each servers' 12 own local RAM 22 a last time for each other server 12 when each other servers' 12 status changed and a value of a status counter at the last time.

Preferably, the reading step includes the step of determining by each server 12 which of the other servers 12 are operating by declaring that each of the other servers 12 whose status has changed within a last predetermined time period is operating. The reading step preferably includes the step of voting by the servers 12 that are up for a winning server 12 that is up and believes it is the winning server 12. Preferably, the reading step includes the step of voting for the server 12 that is up and has a lowest index if no server 12 believes it is the winning server 12.

In the operation of the invention, the system 10 is used to allow N servers 12 to share and arbitrate for access to one or more disk drives. The system 10 works by having N servers 12 perform a voting protocol, using a set of N disk blocks 18 for their communications medium. The servers 12 are all connected to the set of disks 14 for which arbitration is being performed, so

that any server 12 can access all of the disks 14. In a fibrechannel switched network system, the system topology is shown in figure 2.

Note that the server 12 that has won the arbitration election is clear, while the others are cross-hatched in figure 2, indicating that they will not attempt to access the disk drives beyond performing the reads and writes required by the voting protocol.

Note that the server that has won the arbitration election is clear, while the others are cross-hatched, indicating that they will not attempt to access the disk drives. The server that wins the election is called the primary server. Note that this is a dynamic concept, based upon the entity that wins the election, not based upon a fixed assignment.

Note that the box labeled "fibrechannel switched network" is not an arbiter, it is a simple disk controller having several access ports. The arbitration is accomplished by the voting algorithm described here, in that the winner of that election is the only server that is permitted access to the disks.

The server 12 that wins the election is given exclusive access to the relevant set of disks 14. At the time the server 12 transitions from the state of having not won an election to one of having one an election, it must invalidate all of its caches, since during the time between the last time the server 12 won an election and its winning this election, other servers 12 may have modified the contents of the disks 14 in question.

The system 10 must be used with write-through caching, so that at the time that after a server 12 crashes, the disks 14 have

up-to-date data, and a newly elected server 12 can immediately take over responsibility for the set of arbitrated disks 14.

The voting protocol executed by each of the N servers 12 is now described. The system 10 described herein reserves one disk block 18 for each of the N servers 12, at a fixed location on the disk. Each of the N blocks has the following format:

```
struct voteBlock {
        uint32_t heartbeat;
        uint32_t vote;
        uint32_t PrimaryServer;
}
```

Each server 12 is also assigned a fixed index (1..N) such that all servers 12 agree upon which server 12 has which index. This would typically be done when configuring the system 10. This assignment gives all servers 12 a common agreed-upon fixed ordering as determined by this index.

Each server 12 executes the same state machine 20, parameterized by the time constants tHeartbeat, tMin and tMax described below. Once each tHeartbeat seconds (typically 1 second), each server 12 reads all of the vote blocks, and writes its own vote block. There are two other constants, tMin and tMax, (again, typically 10 and 15 seconds, respectively), which will be used below.

Every time that a server writes its own vote block, it increments the heartbeat field, which is a simple counter. The server also places in the vote field the index of the server for whom it is voting; a value of 0xFFFFFFFF indicates that no vote is

being cast. Finally, the amPrimaryServer field contains a 1 if the server believes that it has won the election, and 0 otherwise.

Each server 12 maintains in local RAM 22, for each other server 12, the last time at which the server's heartbeat changed, and the value of the heartbeat counter at that time. The server 12 then determines which of its peers are actually operating by declaring that any site whose heartbeat has changed during the last tMin seconds is up, and otherwise that the server 12 is down.

Figure 3 shows which disk blocks 18 are written and read by which servers 12 in a three server 12 configuration. An arrow from left to right indicates that the server 12 on the left writes the disk block 18 on the right, and an arrow drawn from right to left indicates that the server 12 on the left reads the disk block 18 on the right.

The server 12 determines for whom to vote based on the following algorithm. First, if a proper majority of those servers 12 that are up are voting for a server 12, and that server 12 believes that it has won the election, then the server 12 deciding how to vote casts its vote for the current election winner, to minimize disruption to the election process if servers 12 come up at varying times. Otherwise, the server 12 votes for the working server 12 with the lowest index, subject to the constraint that a server 12 can not vote for a new server 12 within tMax seconds of casting a vote for another server 12; in the interim it must vote for a null server 12 indicating that it casts no vote. If a server 12 discovers that it has a majority of votes from the set of servers 12, it sets the amPrimaryServer flag in its own record to indicate that the server 12 believes that it has won the election at least for another tMin seconds.

Under the assumptions that clock rate skew is limited to being below the fraction (tMax-tMin)/tMax, this algorithm guarantees that no two servers 12 ever simultaneously claim to win an election.

Figure 4 shows the states that the server 12 participating in the election protocol goes through.

In this state machine 20, the transition from state 1 to state 2 occurs if the server 12 has been in state 1 for a minimum of tMax seconds, and during that time, has seen itself as the server 12 that is both functioning and has the lowest index. A transition from state 2 to state 3 occurs after a server 12 sees that it has the votes of a proper majority of the servers 12, including itself. A transition from state 1 to state 4 occurs after a minimum of tMax seconds if the server 12 sees that another server X is the server 12 that is both functioning and has the lowest index of all functioning servers 12. A transition from state 4 to state 5 occurs if the server 12 sees that server X has declared itself the winner of the election. A transition directly from state 1 to state 5 occurs if a minimum of tMax has passed and during that time server X has declared that it has won the election. A transition from state 5 to state 1 occurs if the server 12 that has declared it has won the election no longer appears up (has not updated its heartbeat field for at least tMax seconds), or no longer declares that it has won the election. A transition from state 4 to state 1 occurs if server X no longer appears up. A transition from state 2 to state 1 occurs if we have not yet won the election, and another working server 12 appears with a lower index than ours. A transition from state 3 to state 1 occurs if it sees another server 12 with a lower index decide that it has won the election (this should never happen under normal operation, and

should be logged to an error manager as a significant error in the invention's implementation).

Although the invention has been described in detail in the foregoing embodiments for the purpose of illustration, it is to be understood that such detail is solely for that purpose and that variations can be made therein by those skilled in the art without departing from the spirit and scope of the invention except as it may be described by the following claims.